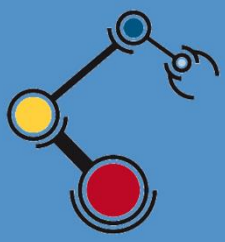
Escuela Politécnica Superior

Control de velocidad y dirección de un vehículo terrestre autónomo



Máster Universitario en Automática y Robótica

Trabajo Fin de Máster

Autor:
Miguel Ángel Contreras Ribera

Tutor/es:
Francisco Andrés Candelas Herías

Julio 2018

Universitat d'Alacant
Universidad de Alicante

Justificación y objetivos

La navegación autónoma de robots es un campo de investigación de gran interés por el amplio número de aplicaciones que se pueden desarrollar en base a esta capacidad, como por ejemplo, la conducción autónoma de vehículos de carretera, reparto de mercancía o actuación en entornos de desastre, como recientemente se pudo observar tras el accidente de Fukushima. Para que robots de estas características puedan operar con seguridad en entornos no estructurados deben disponer de sistemas de visión artificial, LIDAR o GPS, además de un sistema de odometría robusto integrado en el control de velocidad y de dirección.

El objetivo de este trabajo ha sido desarrollar un sistema de odometría de bajo nivel a partir de un sensor de efecto Hall que mide la velocidad del motor en un robot móvil terrestre con geometría Ackermann. En primer lugar, se ha validado la solución escogida, tanto en pruebas en el laboratorio como al aire libre. También hubo que diseñar una pieza de soporte para instalar el sensor en el vehículo y otra para sujetar los imanes que giran solidarios al eje de tracción del robot. Posteriormente, se ha diseñado un sistema de control para la velocidad y otro para la dirección del robot.

Agradecimientos

El presente trabajo ha sido realizado bajo la supervisión del profesor Francisco Andrés Candelas Herías, a quien agradezco su tiempo y sus consejos para que este proyecto haya salido adelante con éxito. Gracias a los estudiantes de doctorado Iván del Pino y Miguel Ángel Muñoz y a los estudiantes de grado Saúl Cova y Yoinel Novelo que colaboran con el grupo AUROVA de la Universidad de Alicante por su dedicación dando vida a BLUE.

También quiero agradecer a mis compañeros del máster su apoyo y amistad, os deseo lo mejor tanto a nivel personal como profesional.

Por último, agradezco a mi madre, a mis hermanas y al resto de mi familia todo su afecto y el apoyo recibido. Habéis conseguido que no pierda la esperanza en ningún momento.

A todos, muchas gracias.

Índice general

Capítulo 1

Introducción

La robótica móvil terrestre es posiblemente uno de los campos de la robótica donde mayor desarrollo puede producirse en unos años, especialmente en el ámbito industrial, donde seres humanos y robots colaboran en tareas como la manipulación y el transporte de mercancías. Para que máquinas de estas características puedan operar con seguridad en entornos no estructurados deben disponer de métodos de estimación de la posición, como los mostrados en la figura 1.1, empleando diferentes tipos de sensores.

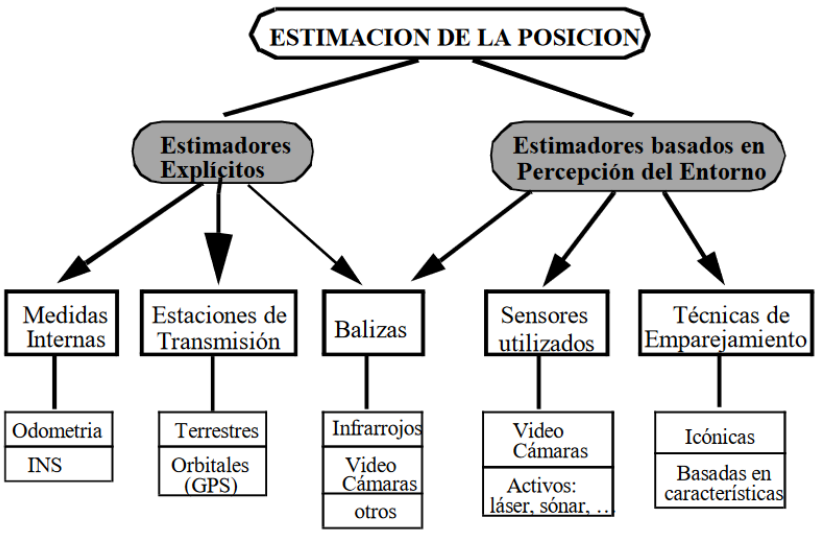


Figura 1.1: Métodos de estimación de la posición [González-Jiménez and Ollero, 1996]

Estos sensores se pueden clasificar de muchas maneras, aunque la más común es en proprioceptivos y exteroceptivos. La propriocepción es la per-

cepción del estado interno del robot, independientemente del entorno, mientras que la exterocepción es la percepción de aspectos externos al robot como la temperatura, la presión o la localización de objetos. Al primer grupo pertenecen los odómetros o las IMUs (Inertial Measurement Unit), mientras que al segundo grupo pertenecen los sistemas GNSS (Global Navigation Satellite System, como sistemas de visión artificial o LIDAR (Light Detection and Ranging) que capturan información del entorno directo del robot.

Este trabajo se centra en el diseño y la validación de un sistema de odometría y un sistema de control para la velocidad y otro para la dirección en un robot móvil con geometría Ackermann.

1.1. Motivación y justificación

El presente proyecto se planteó a partir de la necesidad de realizar la odometría del robot móvil terrestre BLUE [del Pino et al., 2018a] a partir de la velocidad de su eje trasero. El método más habitual para conocer la velocidad de giro de un motor es usar un encoder acoplado al eje del mismo. Al no ser posible acceder al eje del motor, se optó por emplear un sensor magnético de efecto Hall fijado al chasis del robot y un soporte con imanes que gira solidario con el eje trasero del robot. Para medir el ángulo de dirección en el eje delantero se ha empleado un encoder incremental acoplado al eje del servomotor y con una resolución de 1024 pulsos por vuelta.

Además de la odometría, se ha realizado el control PID del motor y del servomotor de la dirección del robot. Inicialmente, se determinaron experimentalmente las ganancias del controlador basándose en un método de ajuste experimental [Ziegler and Nichols, 1942]. Debido a la baja resolución del sensor empleado para medir la velocidad, los efectos del ruido en el ajuste del

Figura 1.2: BLUE en el campus de la Universidad de Alicante

PID hicieron necesario implementar un filtro de Kalman [del Pino et al., 2018b] en la entrada del controlador, empleando los datos del GNSS como referencia.

1.2. Estructura

El trabajo está organizado en seis capítulos, siguiendo el libro de estilo de la Escuela Politécnica Superior de la Universidad de Alicante.

En el capítulo 2 se aborda el estado del arte, es decir, los conceptos y tecnologías básicas en los que se fundamenta el proyecto. En el capítulo 3 se explican los objetivos generales y específicos del trabajo. En el capítulo 4 se describe la metodología empleada, incluyendo las herramientas de hardware y software que se utilizaron. En el capítulo 5 se detalla el desarrollo experimental que se ha llevado a cabo. Por último, en el capítulo 6 se discuten los resultados obtenidos en estos experimentos y se concluye mencionando los trabajos publicados en congresos y revistas científicas, así como futuras líneas de investigación.

Capítulo 2

Estado del arte

En este capítulo se analizan los conceptos y tecnologías que han servido de base para realizar el proyecto. En primer lugar se explican los sensores que forman parte del sistema de odometría del robot. A continuación, se habla de la geometría de Ackermann y se explica el papel que desempeña el módulo CLEAR (Control Logic for Easy Ackermann Robotization) en el control de bajo nivel del robot.

2.1. Encoders o codificadores rotatorios

Para medir la velocidad angular de ejes de maquinaria se emplean, en la mayoría de los casos, unos sensores llamados encoders de rotación o codificadores rotatorios. Estos dispositivos convierten el paso angular en señales digitales o analógicas.

2.1.1. Clasificación

Los encoders rotativos se clasifican en dos tipos: absolutos e incrementales (relativos). Un encoder absoluto envía un código digital de N bits que representa el valor de la posición de su eje, mientras que un encoder incremental envía una señal de pulsos a medida que el eje avanza o retrocede, lo que permite conocer la variación de posición.

Habitualmente, muchos motores eléctricos permiten acoplar un encoder rotativo al eje. En estos casos, se trata de codificadores basados en detectores

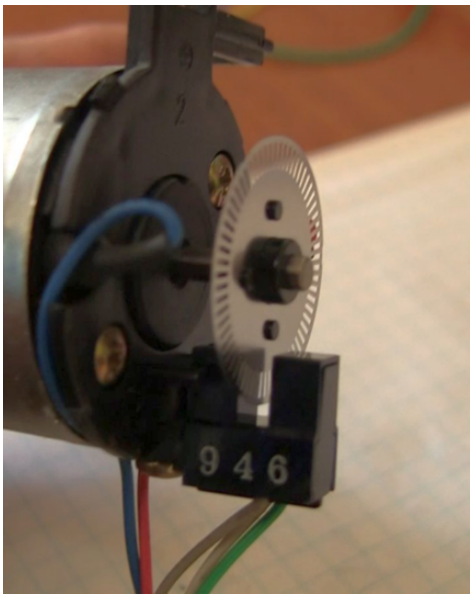


Figura 2.1: Encoder óptico con optoacoplador

ópticos, que pueden usar detectores ópticos de infrarrojos u optoacopladores que miden el giro de un disco perforado, como se muestra en la figura 2.1. Al realizar la medida sin contacto, es un método robusto y duradero. Los encoders rotativos ópticos pueden ser tanto absolutos como incrementales. El motor utilizado en las pruebas en el laboratorio tenía un codificador incremental con un fototransistor.

Si no se dispone de acceso al eje del motor, es posible utilizar codificadores basados en detectores de efecto Hall. Este tipo de sensor es útil como codificador incremental. Al realizar la medida sin contacto, es también robusto y duradero, aunque es sensible a campos magnéticos externos, si el codificador no está bien aislado.

Para medir la velocidad de desplazamiento, el robot BLUE sobre el que se ha realizado este trabajo, utiliza un codificador rotatorio incremental con un detector Hall estándar. Para el sistema de dirección emplea un encoder incremental cuyas características se muestran en la tabla 2.1.

MARCA	Motor Power Company
PART NO.	91470253
CANALES	A y B
RESOLUCIÓN	1024 PPR
ALIMENTACIÓN	8V – 24V
SALIDA	Colector abierto NPN
CONEXIONES	VCC, GND, A y B

Tabla 2.1: Especificaciones del encoder de la dirección

2.1.2. Funcionamiento

El sensor de efecto Hall sobre el que se ha realizado el trabajo tiene una salida o canal por el que entrega una señal de pulsos. Esta salida debe conectarse a un módulo que lleve la cuenta de la posición actual. Estos sensores no permiten determinar el sentido de giro o desplazamiento por sí solos, ni una posición inicial o de referencia.

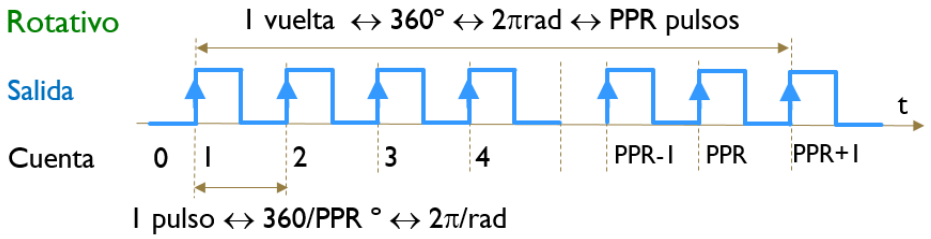


Figura 2.2: Pulsos de un codificador rotatorio

En el caso de los codificadores rotatorios, la medida de posición para cada pulso puede obtenerse a partir de la resolución. El desplazamiento angular R puede expresarse como:

$$R = \frac{2\pi}{PPR}$$

El ángulo recorrido αes por lo tanto:

$$\alpha = Cuenta \cdot R = Cuenta \cdot \frac{2\pi}{PPR}$$

La frecuencia de trabajo f se puede expresar como:

$$f = PPR \cdot \frac{\omega}{2\pi}$$

2.2. Geometría de Ackermann

La geometría de Ackermann es el mecanismo de dirección presente en los automóviles convencionales. La carretilla sobre la que se ha desarrollado el trabajo cuenta con dos ruedas motorizadas en la parte trasera, y dos ruedas directrices en la delantera. En una curva, el radio de giro de las ruedas internas y externas con respecto del centro instantáneo de rotación o ICR es diferente, como se ve en la figura 2.3. Como la velocidad lineal es tangente a la curva, se tienen también velocidades diferentes en cada rueda, lo que evita el deslizamiento. Al estar impulsada a través de su eje trasero, la carretilla lleva un diferencial para trazar curvas sin someterlo a torsión. Las fórmulas para calcular la velocidad para cada una de las ruedas del vehículo son:

$$v_{LR} = \frac{(R-d) \cdot v_D}{R}$$

$$v_{RR} = \frac{(R+d) \cdot v_D}{R}$$

$$v_{LF} = \frac{\sqrt{(R-d)^2 + l^2} \cdot \tan\phi}{l} \cdot v_D$$

$$v_{RF} = \frac{\sqrt{(R+d)^2 + l^2} \cdot \tan\phi}{l} \cdot v_D$$

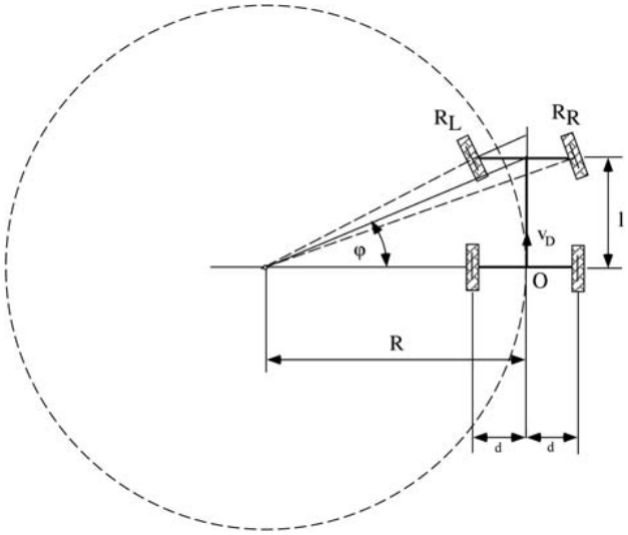


Figura 2.3: Geometría de Ackermann [Berns and Puttkamer, 2010]

En la figura 2.3 se observa la cinemática de un vehículo con geometría de Ackermann. Al contrario que los robots diferenciales, los robots con geometría de Ackermann tienen un radio de giro mínimo distinto de cero [Siegwart et al., 2011]. Esto condiciona el uso de los robots con geometría de Ackermann únicamente a exteriores. En el caso de BLUE, el radio de giro mínimo es 4m aproximadamente.

2.3. La plataforma BLUE

Aunque existen simuladores potentes como Gazebo [Koenig and Howard, 2004] o V-REP [Rohmer et al., 2013] para investigar la navegación autónoma de robots móviles terrestres, los experimentos reales son esenciales para evaluar los resultados y crear aplicaciones reales. Una plataforma autónoma bien diseñada, confiable y fácil de usar y de depurar reduce la cantidad de tiempo requerido para las sesiones experimentales, aumentando la productividad de la investigación.

Actualmente, hay plataformas experimentales con geometría Ackermann como los modelos RB-SHERPA [Robotnik Automation, 2018a] y RB-CAR [Robotnik Automation, 2018b] fabricados por Robotnik que se muestran en la figura 2.4.

Figura 2.4: Plataformas RB-SHERPA y RB-CAR

Después de valorar diferentes opciones, el grupo AUROVA decidió desarrollar un robot terrestre derivado de un carro eléctrico de uso industrial. BLUE ha sido adaptado para ser controlado por software e integrado con ROS. Además, la arquitectura del sistema permite implementar gradualmente las diferentes capas de abstracción obteniendo diferentes capacidades: control teleoperado ROS, recopilación de datos para investigación SLAM o un sistema autónomo que incluye una estación base y uno o más rovers.

2.4. El módulo CLEAR

Debido a que las máquinas con geometría tipo Ackermann comparten características mecánicas, se ha desarrollado un módulo genérico y de código y hardware abierto que contiene los elementos de hardware y software necesarios para integrar este tipo de vehículos en ROS. Este módulo se llama CLEAR (Control Logic for Easy Ackermann Robotization) y permite con-

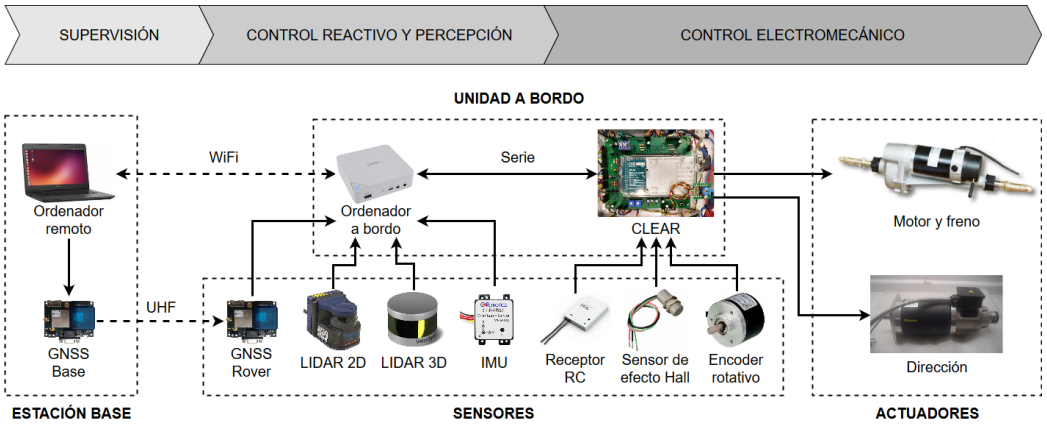


Figura 2.5: Arquitectura del robot BLUE [del Pino et al., 2018a]

trolar la velocidad y la dirección del robot y ajustar dinámicamente todos los parámetros desde una interfaz ROS. Este módulo facilita el proceso de depurado, la experimentación y la creación de datasets, y gestiona la transición entre estados de emergencia, calibración y control desde ROS o desde una emisora RC.

2.4.1. Arquitectura

CLEAR es un *middleware* entre el hardware de la plataforma y ROS. Como se puede ver en la Fig. 2.6, cuenta con dos niveles de abstracción y está compuesto por una electrónica concreta que conecta la capa de bajo nivel (actuadores y sensores) y un software que hace de interfaz con un sistema de alto nivel. De esta forma, CLEAR permite acceder a la velocidad, la dirección y el freno para monitorizar, configurar y controlar el robot. También incorpora una interfaz hombre-máquina que informa del estado del sistema y que puede cortar la corriente en caso de emergencia.

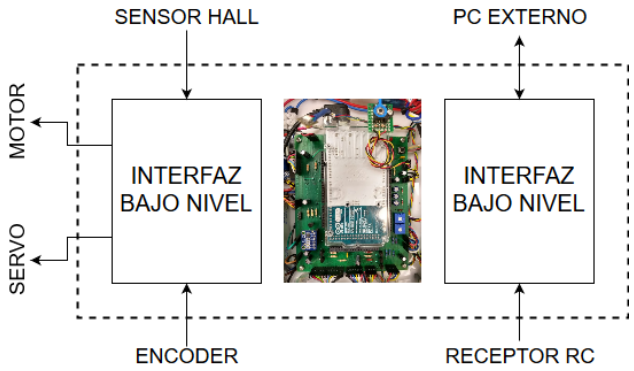


Figura 2.6: Arquitectura de CLEAR [del Pino et al., 2018c]

2.4.2. Software

Para poder integrar el módulo CLEAR en varias plataformas se desarrolló un código genérico, modificable y que permite cambiar de varias formas los parámetros internos necesarios para adaptar el módulo a cada robot. Además de ajustar parámetros de forma dinámica, es posible reorganizar las entradas y salidas del módulo CLEAR modificando un solo fichero. Asimismo, un archivo de descripción del robot permite establecer los valores fijos como dimensiones, distancia entre ejes o su velocidad máxima y el ángulo de giro máximo del robot. También es posible ajustar las ganancias del PID a través de un tópico de ROS.

El software de CLEAR se desarrolló en C++, encapsulando los componentes del robot de bajo nivel como objetos de diferentes clases. Estos componentes están dispuestos en una estructura de árbol, con la clase *Vehicle* como raíz principal. Gracias a esto, el valor de las variables se propaga fácilmente de más bajo a más alto nivel, lo que facilita el proceso de depuración, minimizando los errores que puedan surgir al modificar el código y evitando que se propaguen a otros sistemas.

2.4.3. Interfaz de alto nivel

ROS proporciona una forma sencilla y robusta de comunicaciones entre distintos nodos [Quigley et al., 2009]. CLEAR está encapsulado en un nodo comunicado por *rosserial* y a través del cual viajan tres tipos de tópicos distintos:

- **Tópicos de control**

  Permiten enviar los estados Ackermann deseados y las ganancias de los controladores PID. A través de otro tópico *echo* también se puede configurar el nivel de información que envía el vehículo para el depurado.

- **Tópicos de depurado**

  Permiten visualizar los valores de los comandos que se han enviado a través de los tópicos de control para verificar que internamente está funcionando correctamente.

- **Tópicos de monitorización**

  Permiten visualizar variables como las medidas de los encoders, las entradas y salidas de los controladores PID o la configuración actual del sistema.

2.4.4. Interfaz de bajo nivel

La capa de control de bajo nivel de CLEAR está preparada para emplear codificadores para estimar la velocidad y el ángulo de giro del robot. Como acción de control usa un controlador PID para cada sistema. Al haber empleado un sistema de odometría de baja resolución se consideró implementar dos variantes del filtro de Kalman. La primera es un filtro del Kalman unidimensional (SDKF) que utiliza la señal de control para alimentar la etapa

de predicción del filtro y reduce el modelo a una variable. Esta aproximación es válida siempre que el posicionamiento de la dirección sea conocido. Si no fuera posible, se emplea el filtro de Kalman extendido (EKF) de 3 dimensiones para estimar la velocidad, la variación del ángulo de giro y el ángulo inicial. Se asume que hay una relación lineal entre la tensión del motor y la velocidad [del Pino et al., 2018b] y que será diferente para cada plataforma.

2.4.5. Hardware

El núcleo de CLEAR es un Arduino MEGA 2560, una placa de hardware libre para la realización de prototipos electrónicos con un microcontrolador ATmega2560 de 8 bits. Todos los circuitos principales de CLEAR que trabajan a 5V están organizados en la misma PCB principal (CLEAR PCB), mientras que los acondicionadores de señal para los sensores, controladores y relés para los actuadores, están en PCBs diferentes, por seguridad y facilidad de sustitución.

Para la interfaz de sensores se desarrollaron otras placas con acondicionamiento aislado para señales digitales de detectores y codificadores NPN o PNP mediante optoacopladores. Además, CLEAR incluye un LED RGB de alta luminosidad para mostrar el estado del vehículo y está conectado a un receptor DR16 de DJI que permite controlar el vehículo con una emisora RC.

Como ya se ha mencionado, la placa CLEAR está diseñada para interactuar con el codificador de dirección y el de odometría y está habilitada para contar pulsos por medio de interrupciones.

Durante los experimentos en el robot BLUE, se detectó que el Arduino perdía pulsos porque no era capaz de gestionar las interrupciones para el codificador de alta resolución del sistema de dirección. Para corregir este problema, fue necesario incorporar una placa Teensy 3.2 con un microcon-

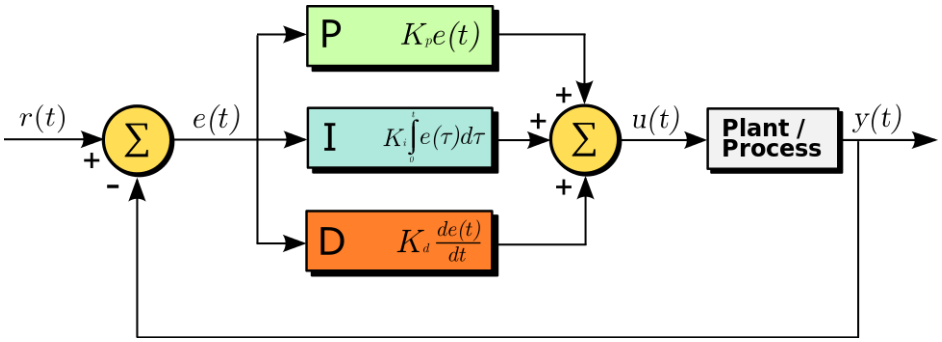


Figura 2.7: Controlador PID en lazo cerrado [PID controller, 2018]

trolador de 32 bits. Esta placa se comunica con CLEAR a través del bus I2C como dispositivo esclavo.

2.5. Control PID

En este apartado se hace una introducción a la teoría sobre control PID, el cual se ha empleado en el control de la velocidad y la dirección. A pesar de ser un método antiguo, los controladores PID son útiles debido a que se pueden aplicar a la gran mayoría de sistemas de control [Ogata, 2010].

En la figura 2.7 se tiene representado un controlador PID genérico. Al ser un sistema de lazo cerrado o realimentado, se toma una medida de la salida *y(t)* y se compara con una señal de entrada conocida como consigna *r(t)*. Al restar la medida de la salida a la consigna, se obtiene el error *e(t)* o señal de realimentación.

Un controlador PID está compuesto por tres componentes (proporcional, integral y derivativa) cuyo comportamiento se describe a continuación:

- **Componente proporcional (P)**

  La acción de control es proporcional al error. Incrementando la ganancia proporcional K_p se incrementa la velocidad del sistema y disminuye el

error estacionario, pero si la K_p se incrementa demasiado la variable del proceso comenzará a oscilar, haciendo el sistema inestable.

- **Componente integral (I)**

  Integra el error en el tiempo para minimizar el error en estado estacionario. La respuesta integral continúa incrementando con el tiempo a menos que el error sea cero. Si la ganancia integral K_i es demasiado pequeña puede haber sobreoscilaciones o problemas de inestabilidad.

- **Componente derivativa (D)**

  Anticipa el comportamiento futuro del error, ya que la componente derivativa es proporcional a la tasa de cambio del error, y de esta forma ayuda a evitar que se produzcan oscilaciones. La ganancia derivativa K_d suele ser pequeña porque la respuesta derivativa es muy sensible al ruido en la señal de la variable del proceso.

En las pruebas iniciales con el robot levantado sobre el gato, se empleó una librería PID genérica para Arduino [Beauregard, 2017]. Posteriormente, esta librería ha sido sustituida por otra específica que se encuentra dentro del repositorio de GitHub de CLEAR [AUROVA, 2018b].

Capítulo 3

Objetivos

Este proyecto nace dentro de la línea de investigación en navegación autónoma terrestre del grupo de Automática, Robótica y Visión Artificial (AUROVA) de la Universidad de Alicante.

Después de evaluar las especificaciones de las plataformas robóticas existentes se llegó a la conclusión de que lo más apropiado sería desarrollar una nueva plataforma a partir de una carretilla eléctrica de uso industrial Zallys JESPI. Esta máquina posee una capacidad de carga de hasta 600 Kg y alcanza una velocidad máxima de 5 Km/h, el paso de una persona caminando, y es capaz de subir pendientes de hasta 30 grados de inclinación. A pesar de no tener un sistema de suspensión, la carretilla tiene neumáticos con tacos de goma que la hacen apropiada para su uso en exteriores.

Para adecuar la carretilla a su uso como vehículo autónomo se sustituyó el sistema de dirección original, un timón accionado por el operario, por un servomotor y un encoder de alta resolución. En la figura 3.2 puede verse el servomotor y el sistema de transmisión formado por un piñón y una corona cónicos. También se instaló una plancha de metal sobre el chasis, con agujeros roscados para facilitar la instalación de otros componentes del robot, como el LIDAR o la IMU.

Este trabajo abarca la odometría y el control de velocidad y dirección del robot terrestre de tipo Ackermann basado en la carretilla mencionada anteriormente, desarrollado como plataforma de investigación para tareas de navegación autónoma. Para ello, se han realizado experimentos para validar

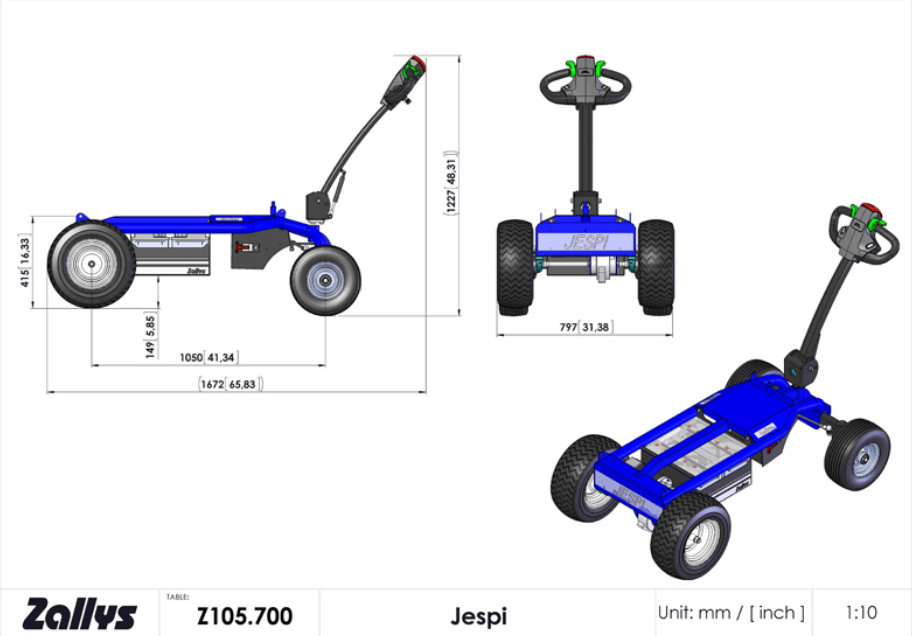


Figura 3.1: Dimensiones de la carretilla Zallys JESPI

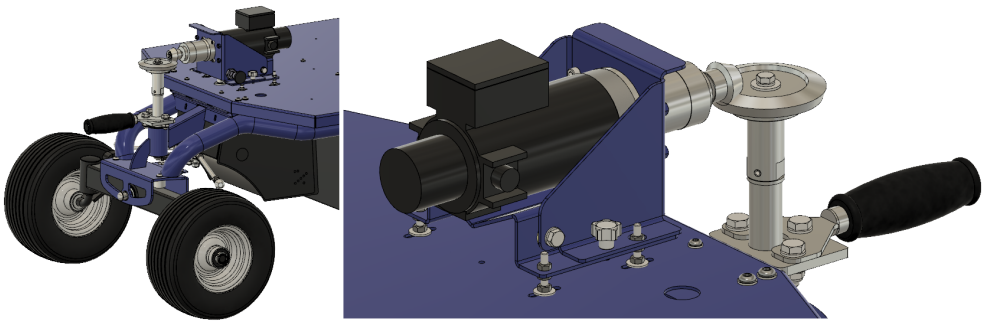


Figura 3.2: Modificaciones realizadas en la dirección

la solución escogida y poder detectar posibles fallos debidos, por ejemplo, a un exceso de ruido en el sensor. También se han diseñado las piezas de soporte, fabricadas en PLA con una impresora 3D, y que después se montaron en el vehículo. A partir del sistema de odometría, se realizaron experimentos en exteriores comparando la velocidad dada por el sensor Hall con la velocidad dada por el GNSS.

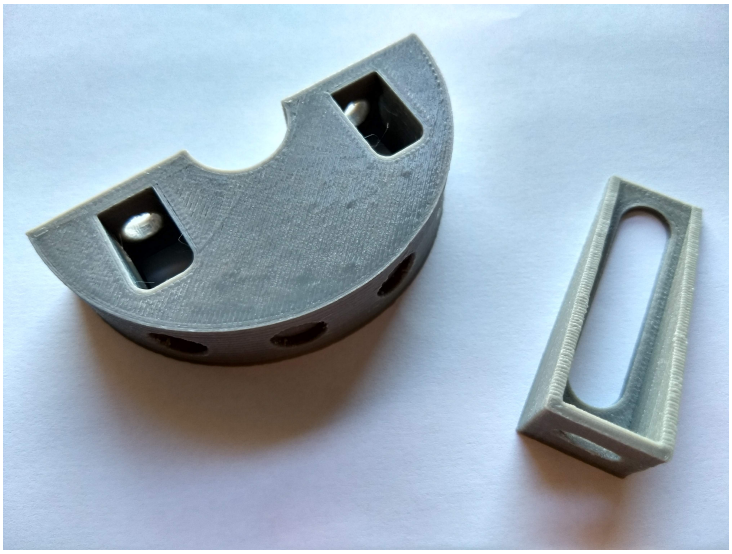


Figura 3.3: Piezas diseñadas ya impresas

Se trata de un trabajo multidisciplinar, que ha servido para dotar al robot de una medida de velocidad fiable sobre la que diseñar un sistema de control (en velocidad, aceleración o jerk) junto con el sistema de control (en posición o velocidad) para la dirección del robot. Ésto servirá de base para futuros trabajos de investigación sobre navegación autónoma.

Capítulo 4

Metodología

En este capítulo se describen las técnicas y las herramientas que se han utilizado a la hora de evaluar el sistema de odometría propuesto y diseñar un sistema de control para el motor y el servo de dirección del robot.

4.1. Odometría

Para realizar el sistema de odometría de este trabajo, se tuvieron en cuenta las limitaciones dadas por la geometría del vehículo. El diámetro de la pieza que sirve de soporte a los imanes ha estado condicionado por la distancia entre el eje y la chapa y también por el diámetro de éstos.

Medir la distancia entre dos partes poco accesibles usando una cinta métrica no es fácil ni preciso. Al disponer de un modelo CAD de la carretilla modificada, fue posible trazar una línea de cota entre el eje y la chapa y utilizar este valor como referencia para el diámetro de la pieza, como se muestra en la figura 4.1. Para ello, se empleó el programa Fusion 360 de Autodesk, con el que se diseñaron también las piezas.

Para medir el giro de la pieza se escogió un sensor Honeywell de la serie 103SR con una salida digital. En el robot este sensor está alimentado a 12V aunque admite tensiones desde 4.5V hasta 24V. El sensor cuenta con una carcasa de aluminio, por lo que es apropiado para ser usado en ambientes no corrosivos.

En la figura 4.2 se ha mostrado un diseño de la pieza de soporte de los

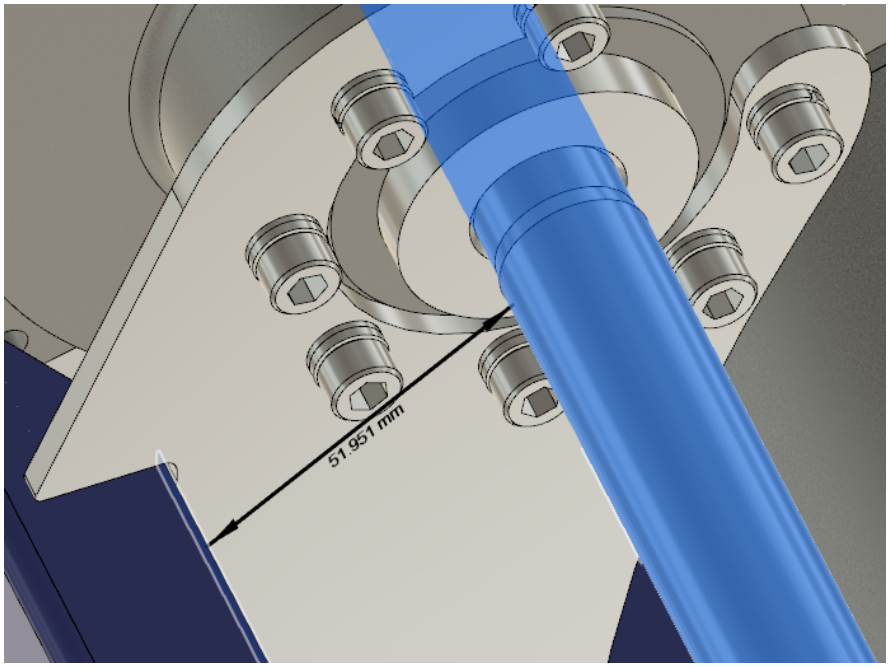


Figura 4.1: Cota entre el eje y la chapa

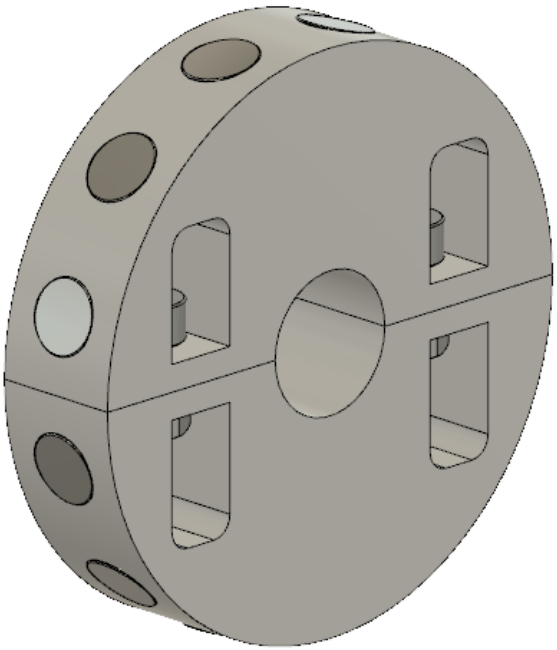


Figura 4.2: Ensamblaje de la pieza montada en el eje trasero del robot

Series	103SR (digital)
Description	Hall-effect digital position sensor
Package material and style	aluminum threaded barrel
Magnetic actuation type	unipolar, bipolar, latching
Operation	proximity to external magnet
Supply voltage range	4.5 Vdc to 24 Vdc
Supply current	4 mA to 10 mA (inclusive)
Output type	digital sinking
Operating temperature range	-40°C to 100°C [-40°F to 212°F]
Dimensions	Ø11,9 mm x 25,4 mm [15/32-2 x 1.0 in]
Features	unipolar, bipolar, and latching magnetics; sinking or sourcing output, aluminum housing, color-coded jacketed cable, adjustable mounting

Figura 4.3: Especificaciones del sensor 103SR13A-1

imanes ya ensamblada. Cuando se produce el paso de un imán, el sensor crea una señal electrónica para comunicarse con el sistema de control y que éste convierte en un valor de velocidad. A este tipo de sensores se les llama magnetorresistivos.

En la figura 4.3 se muestra una tabla con los datos del sensor 103SR13A-1 sacada del catálogo del fabricante.

4.1.1. Banco de pruebas

Para probar el sensor descrito en la sección 4.1 se empleó una placa Arduino, una fuente de alimentación y un motor de corriente continua con un

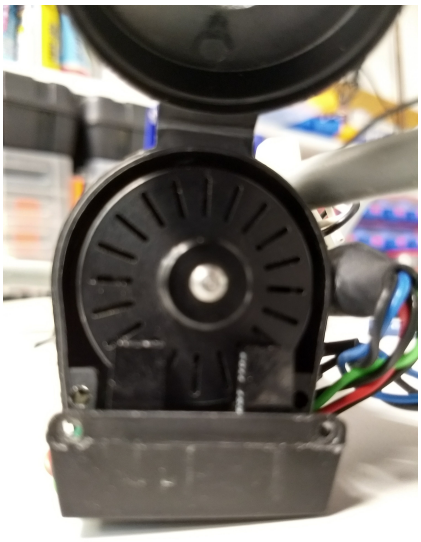


Figura 4.4: Detalle del interior del encoder empleado en las pruebas

encoder. Mientras se alimentaba el motor con tensiones de 24V y 31V se midió el número de vueltas a través del encoder y el sensor de efecto Hall. Ambos sensores estaban alimentados a 5V por la placa Arduino y sus salidas digitales estaban conectadas a entradas digitales de la misma. La placa estaba alimentada por USB desde un ordenador portátil.

Posteriormente se compararon las medidas realizadas por el encoder y el sensor de efecto Hall para demostrar la validez de este tipo de sensores cuando se emplean para medir la velocidad angular en el eje de las ruedas traseras.

4.2. Software

En este apartado se analizan las herramientas y metodologías de programación empleadas en las pruebas que se detallan en el capítulo siguiente.

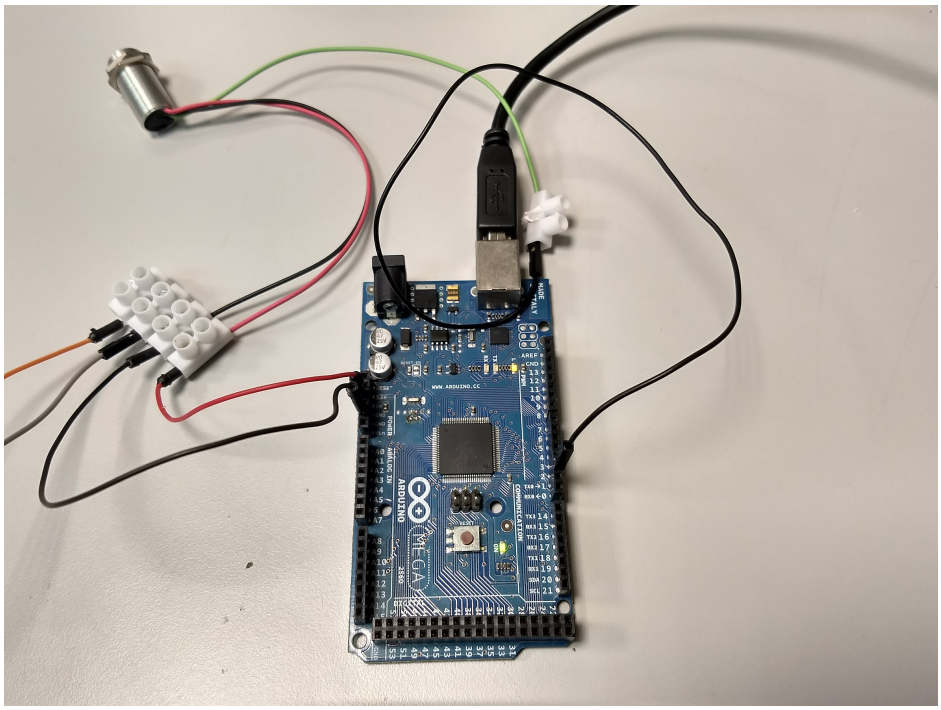


Figura 4.5: Placa Arduino MEGA con los sensores conectados

4.2.1. Editores y entornos de programación

Para programar desde Windows 10 la placa Arduino que se utilizó en las pruebas iniciales y la Teensy 3.2 que lee la señal de los codificadores, se ha empleado el IDE de Arduino disponible en su página web oficial. No obstante, muchos editores de código como VS Code o Atom ofrecen *plugins* o extensiones que permiten programar este tipo de placas. Algunos entornos de desarrollo integrados como Eclipse disponen también de este tipo de extensiones. El proyecto completo de CLEAR se desarrolló en Ubuntu 16.04 LTS utilizando Eclipse CDT con la extensión Sloeber. La versión de ROS con la que se ha trabajado es Kinetic Kame.

Para procesar las medidas que se obtuvieron en los distintos experimentos se ha empleado MATLAB.

4.2.2. Control de versiones con Git

Tanto el proyecto de BLUE como el de CLEAR se han desarrollado empleando el sistema Git de control de versiones. Lo que diferencia a Git de otros sistemas como Subversion, es que este último almacena los datos o *blob* como modificaciones de cada archivo con respecto a una versión base, mientras que Git almacena los datos como un conjunto de instantáneas. Es decir, cada vez que se confirman los cambios o *commits* en Git es como hacer una foto de los archivos en ese instante, y se guarda una referencia o *hash-object*.

Los cambios que se realizan en Git están almacenados localmente en el ordenador. Para poder alojar proyectos de software que emplean este sistema de control de versiones y hacerlos públicos se emplean plataformas web como GitHub, GitLab o Bitbucket. Tanto el código para lanzar los nodos de ROS de BLUE como los archivos que integran la lógica de CLEAR están alojados en repositorios de GitHub [AUROVA, 2018a,AUROVA, 2018b]. Esto permite trabajar con *forks* o autorizar *pull requests* de colaboradores.

Capítulo 5

Desarrollo

Este capítulo se divide en dos partes, en la primera se explican las pruebas que se realizaron para validar el sensor de efecto Hall, y en la segunda se describe el proceso seguido para diseñar el sistema de control.

5.1. Validación del sensor

Alimentando un motor CC con diferentes valores de tensión se midió el número de vueltas utilizando un codificador óptico fiable acoplado al eje del motor y un sensor magnético de efecto Hall al girar una pieza cilíndrica con imanes en la salida de la reductora.

Usando el encoder óptico como referencia o *ground truth* se compararon las medidas recogidas por ambos sensores. Con ésto, se quiso demostrar la validez de emplear un sensor de efecto Hall para medir la velocidad angular en el eje de las ruedas traseras, ya que era imposible utilizar un encoder acoplado al motor de la carretilla.

5.1.1. Banco de pruebas

En la primera parte de la prueba se empleó una pieza fabricada mediante impresión 3D con seis huecos para los imanes y que se acopló en el eje del motor, como se muestra en la figura 5.1.

A partir de los datos de salida obtenidos con el programa de Arduino, se observó que a diferente tensión las medidas del sensor de efecto Hall estaban

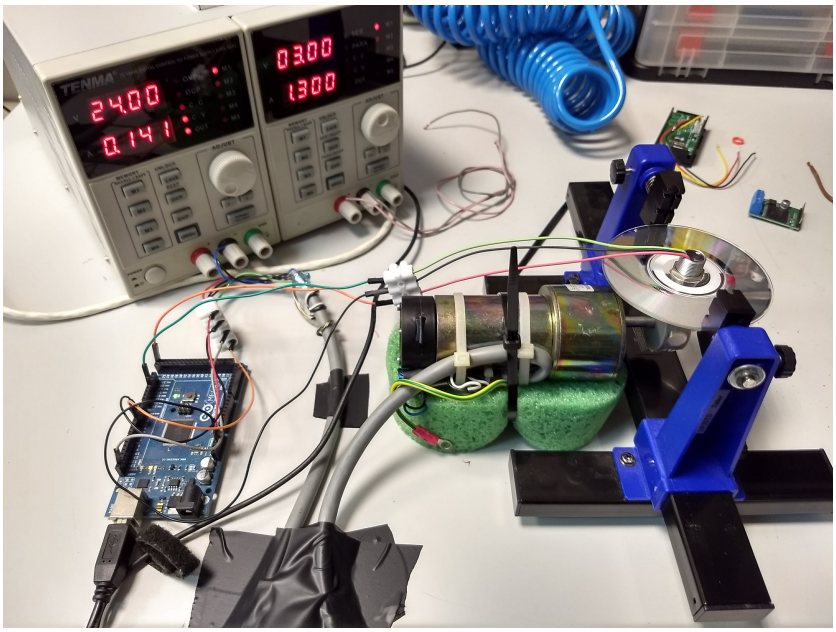


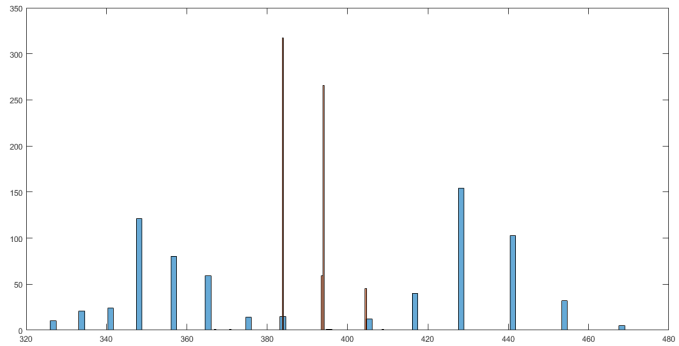
Figura 5.1: Pruebas iniciales en el laboratorio

desviadas con respecto de las medidas recogidas por el codificador óptico.

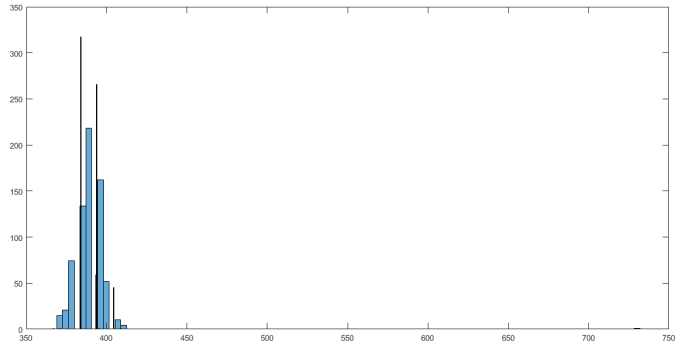
En la figura 5.2 se comparan las medidas del sensor Hall, de color azul, con las medidas del encoder, de color rojo, a 31V empleando distintas frecuencias de refresco. Estas medidas han sido normalizadas para ser comparadas con la media del encoder, como se muestra en la figura 5.3. En las medidas del sensor Hall se observa dispersión y la presencia de *outliers* o valores atípicos muy alejados de la media.

5.1.2. Diseño de las piezas de soporte

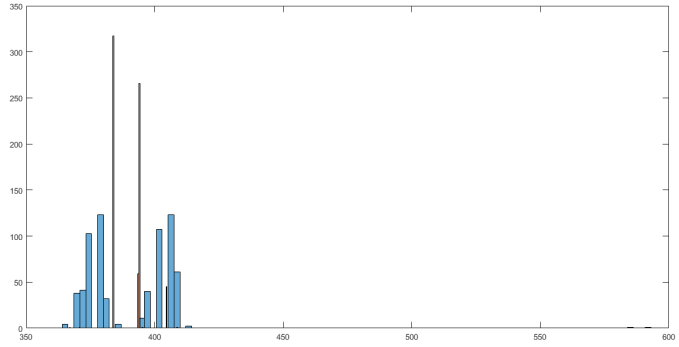
Después de realizar las pruebas con la pieza pequeña de 6 imanes, se diseñó un primer diseño de la pieza que iba a estar colocada en el vehículo. También se diseñó un adaptador para el eje del motor de continua empleado en las pruebas. Como se aprecia en la figura 5.4, la nueva pieza estaba formada por dos mitades iguales con 12 imanes cada una. Se consideró que en caso



(a) Refresco cada pulso

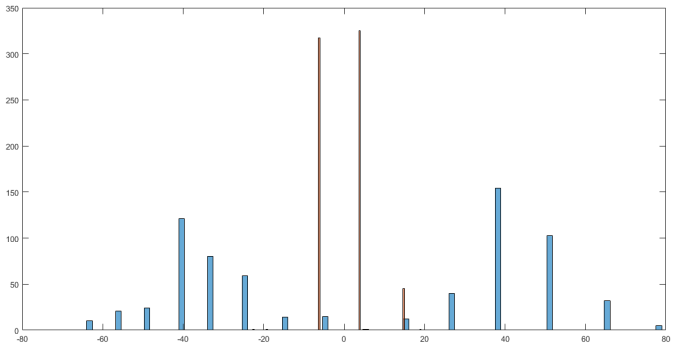


(b) Refresco cada 2 pulsos

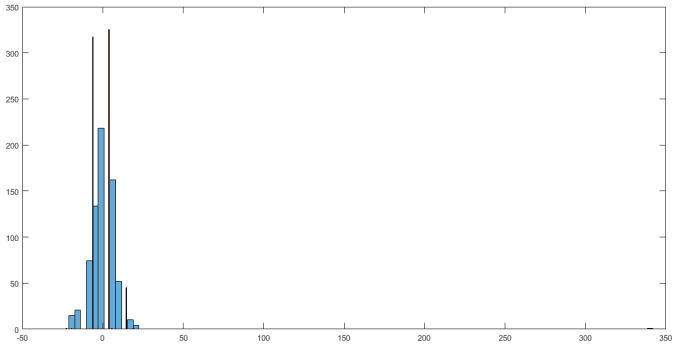


(c) Refresco cada 3 pulsos

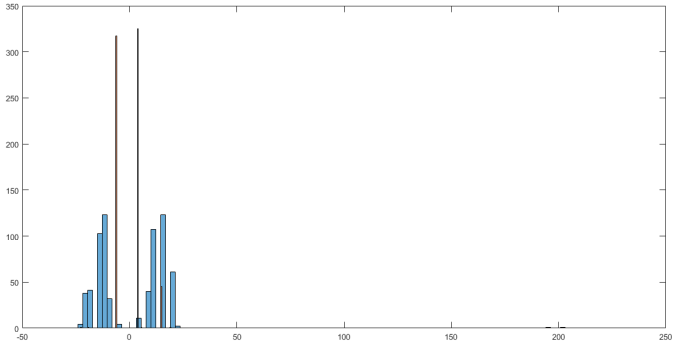
Figura 5.2: Medida de las velocidades del sensor Hall respecto del encoder



(a) Refresco cada pulso



(b) Refresco cada 2 pulsos



(c) Refresco cada 3 pulsos

Figura 5.3: Dispersión de las velocidades del sensor Hall respecto del encoder

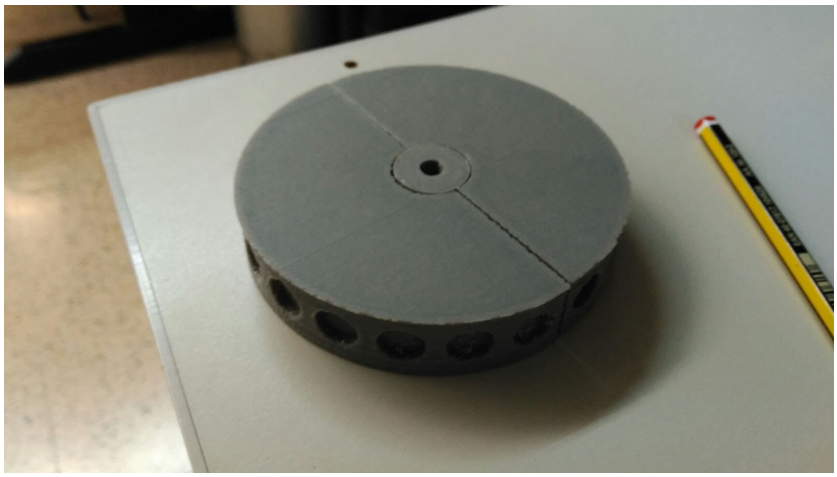


Figura 5.4: Primera versión de la pieza de soporte de los imanes

de romperse una de las mitades, sería más fácil sustituirla. Sin embargo, la pieza se diseñó con muy poca separación entre los imanes y a veces se perdían pulsos.

Después de varias iteraciones, se diseñó una pieza formada por dos mitades con seis ranuras en cada una. Ambas mitades tienen un diámetro de 84.5 mm y están unidas por dos tornillos M6 con cabeza Allen. En esta pieza, los imanes están separados por una distancia igual al diámetro de los mismos y se pegaron a la pieza con pegamento de Cianocrilato.

También se realizó el diseño de la pieza de soporte al sensor Hall. Está reforzado y tiene algo de holgura en la base para asegurarse de que queda alineado con el soporte de los imanes.

5.1.3. Montaje de las piezas

Fue necesario utilizar un gato de tijera para elevar la carretilla y poder quitar la rueda trasera del lado izquierdo. Las dos mitades del soporte de los imanes se unieron con dos tornillos M6 y también se empleó cinta adhesiva para evitar que la pieza deslizara sobre el eje en el que está montada. Para

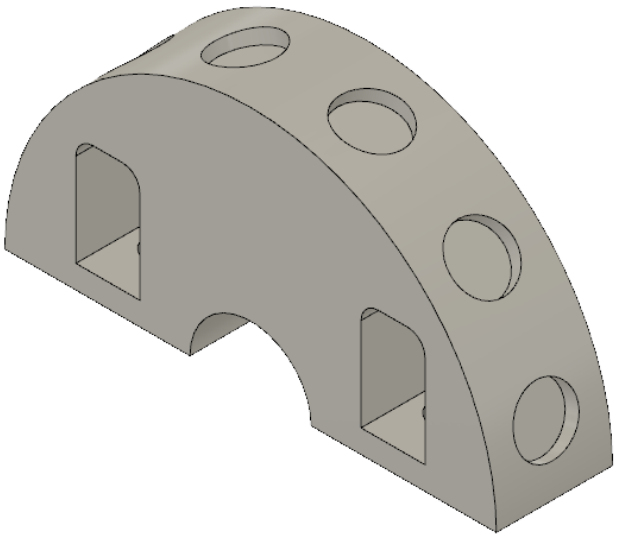


Figura 5.5: Modelo 3D de la pieza de soporte de los imanes

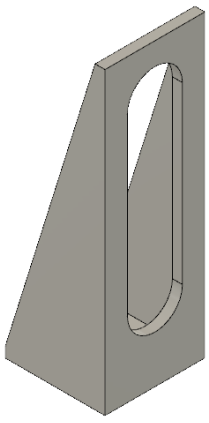


Figura 5.6: Modelo 3D de la pieza de soporte del sensor

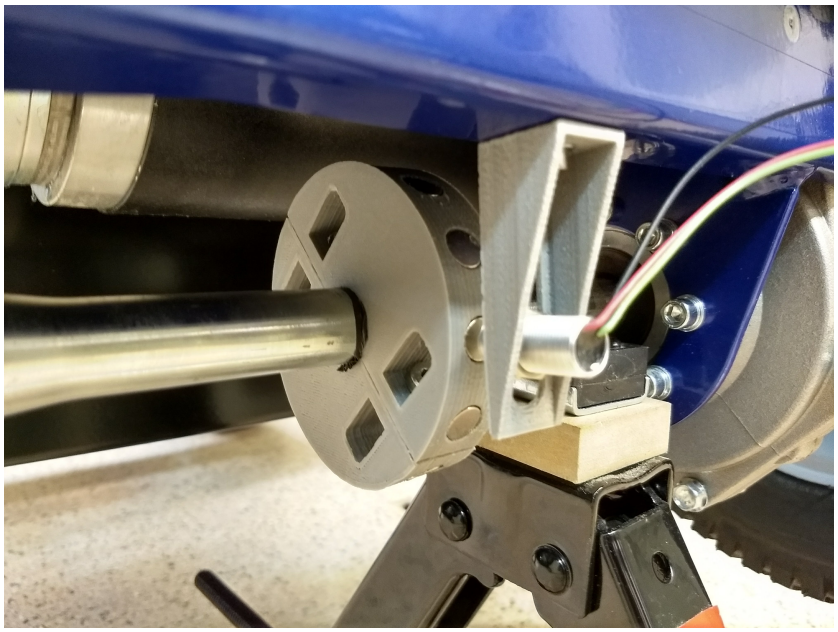


Figura 5.7: Instalación de las piezas y el sensor en el robot

instalar el soporte del sensor se tuvo que hacer un agujero en la chapa del vehículo con un taladro. Después, se acercó la cara del sensor a los imanes, comprobando que estuviesen alineados. En la figura 5.7 pueden verse las piezas ya instaladas en el vehículo.

5.1.4. Pruebas iniciales

Con la pieza y el sensor Hall ya montados y con el robot apoyado en un gato, se realizó una prueba en el interior del laboratorio para estudiar el ruido del sensor Hall comparando la velocidad de las ruedas con la tensión en el motor del robot. Mediante la emisora RC se hacía girar el motor y a través de los tópicos *desired_ackermann_state* y *speed_volts_and_steering_pwm_being_applicated* se registraban los datos de velocidad y tensión respectivamente en dos archivos TXT. A partir de estos archivos se han generado dos archivos CSV en MATLAB. Los resultados se han normalizado dividiendo por los valo-

res máximos de velocidad y tensión, según el caso, y se han representado gráficamente.

En la figura 5.8 se ha representado de color verde la medida de velocidad y de color azul la tensión del motor. Aunque hay correlación entre tensión máxima y velocidad máxima, se observa que el ruido es especialmente problemático con incrementos (y decrementos) de velocidad pequeños, produciendo picos en la lectura.

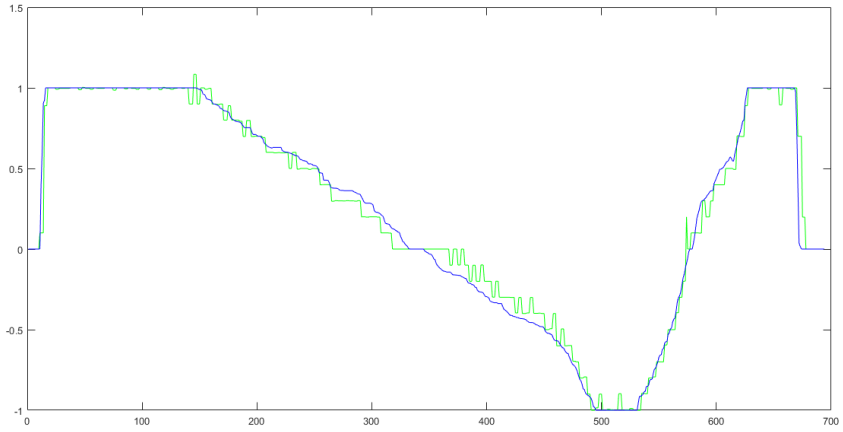


Figura 5.8: Comparación entre tensión y velocidad

5.2. Control del motor y del servomotor

En este apartado se incluyen los experimentos realizados para ajustar el controlador PID de la velocidad del motor. Tomando como consigna el valor normalizado de velocidad a través de la emisora RC, se va a medir la velocidad normalizada del motor leída por el sensor Hall y a compararla con la tensión en el motor normalizada.

5.2.1. Ajuste inicial

Teniendo presente el ruido del sensor observado anteriormente, se consideró emplear un método de calibración empírico bastante habitual [Ziegler and Nichols, 1942], aunque primero se estudiaron los efectos por separado de las ganancias en el motor. En la figura 5.9 se han representado un experimento con controladores P, I y PI. La velocidad deseada se ha enviado a través de la emisora RC y está representada de color rojo, la velocidad medida mediante el sensor de efecto Hall de color verde, y la tensión del motor de color azul.

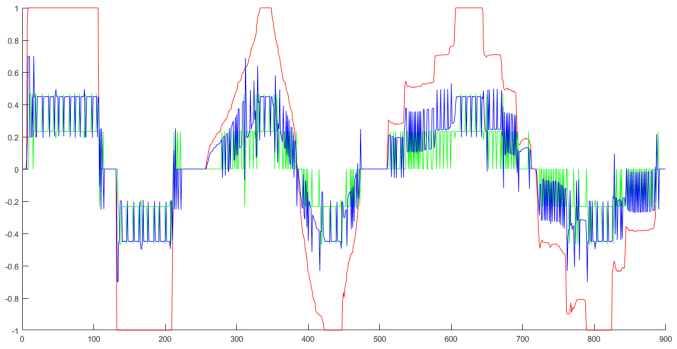
5.2.2. Problemas detectados

Se comprobó que el PI es muy poco estable y hacía que el motor diera tirones. Posteriormente, se probó un controlador PD con $K_p = 0{,}7$, $K_i = 0{,}0$ y $K_d = 0{,}05$ que provocaba sacudidas cuando la consigna de la velocidad era cero. En estas pruebas se vio que por culpa del ruido, solamente es posible hacer ajustes en la K_d del controlador. También se pensó que era necesario aplicar un filtro a la entrada para eliminar todo ese ruido.

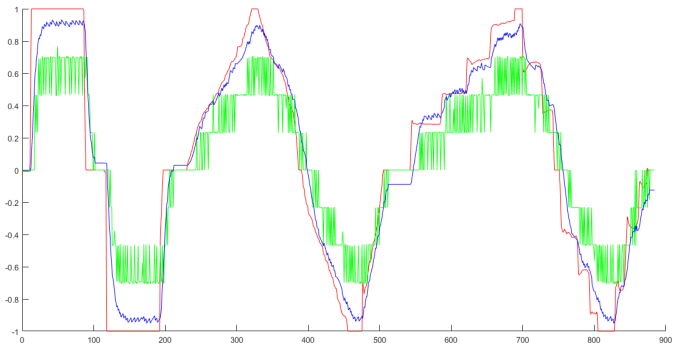
5.2.3. Implementación del filtro

Se decidió probar un filtro de Kalman unidimensional o *single-dimensional Kalman filter (SDKF)* [Shaowei and Shanming, 2012]. Con ello, se buscaba eliminar *outliers* en la aceleración. La implementación del filtro de Kalman no forma parte de este trabajo y ha sido publicada en un artículo para una conferencia internacional [del Pino et al., 2018b].

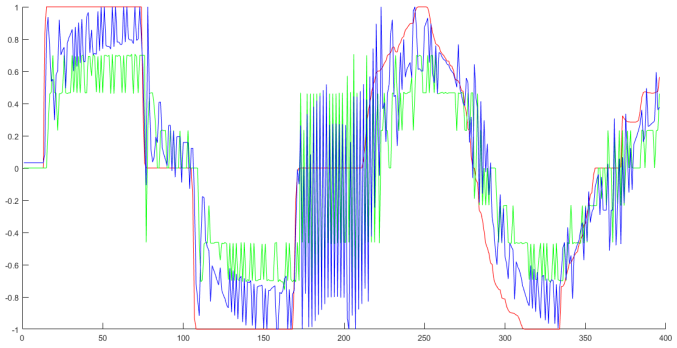
Debido a la alta resolución del codificador rotatorio empleado, se consideró que para la dirección no era necesario el filtro de Kalman. El controlador



(a) $K_p = 0{,}7$



(b) $K_i = 1{,}0$



(c) $K_p = 0{,}7$ y $K_i = 1{,}0$

Figura 5.9: Diferentes ajustes del controlador del motor

del sistema de dirección es PI y las ganancias se obtuvieron experimentalmente, siendo éstas $K_p = 1{,}0$ y $K_i = 3{,}0$.

Capítulo 6

Resultados y conclusiones

En este capítulo se analizan los resultados obtenidos en los experimentos del capítulo anterior y se comentan aspectos del trabajo que pueden tenerse en cuenta en trabajos futuros.

6.1. Resultados

Se ha conseguido diseñar, montar y poner en funcionamiento un sistema de odometría basado en una sensor de efecto Hall, así como medir la velocidad y el ángulo de giro con suficiente precisión. Posteriormente se han programado los controladores PID para la velocidad y para la dirección.

El trabajo realizado en este proyecto, junto con otros desarrollos para la plataforma BLUE, han sido descritos en publicaciones en congresos de prestigio, lo que también valida su interés y utilidad. A continuación se ha elaborado una lista con los trabajos presentados hasta hoy en congresos nacionales e internacionales:

- Presenting BLUE: A roBot for Localization in Unstructured Environments (ICARSC 2018)
- Speed Control of an Unmanned Ground Vehicle Using Extremely Low Resolution Sensors (ICINCO 2018)
- Proyecto BLUE: Robot para Localización en Entornos no Estructurados (Jornadas Nacionales de Robótica 2018)

- CLEAR. Un Módulo para la Robotización de Máquinas Ackermann (XXXIX Jornadas de Automática)

Además, el artículo Presenting BLUE: A roBot for Localization in Unstructured Environments [del Pino et al., 2018a] ha sido seleccionado con extensión a la prestigiosa revista científica *Journal of Intelligent & Robotic Systems (JIRS)* donde será publicado con el título *Deeper in BLUE: Development of a roBot for Localization in Unstructured Environments*.

6.2. Conclusiones

Para conseguir que el control de la velocidad y el control de la dirección sean válidos, se han tenido que hacer muchos ensayos con el hardware y con el software para ajustar el controlador. Esto demuestra que, a pesar de disponer de herramientas de simulación como las mencionadas en el capítulo 2, las pruebas con el equipamiento real son necesarias.

Aunque se ha desarrollado un sistema de odometría robusto y de bajo coste, para realizar un control preciso del motor es necesario implementar algoritmos de filtrado debido a la baja resolución del sensor empleado. Además, al haber empleado un encoder de alta resolución en el sistema de dirección, se necesita más potencia de cálculo para contar los pulsos y estimar la posición angular.

Por otro lado, al haber empleado placas con microcontroladores potentes y baratas, como el Arduino que utiliza CLEAR, ha sido posible simplificar el hardware y disminuir su coste, con el inconveniente de haber complicado el software.

Como se ha visto en el apartado 2.3, el vehículo tiene diferencial en el eje de las ruedas motrices. Se ha comprobado que a lo largo de un recorrido

de prueba, con multitud de giros en ambas direcciones, el error causado al considerar únicamente la velocidad de una única rueda es muy pequeño. No obstante, es posible calcular la velocidad de la otra rueda con las restricciones cinemáticas que se mencionan en el apartado 2.2.

6.3. Trabajos futuros

En un futuro cercano, se quiere que las piezas de soporte del sensor y de los imanes estén hechas de un material durable (metal o resina acetálica). También se valora cambiar el encoder incremental del sistema de dirección por uno absoluto, o en su defecto, añadir un potenciómetro que ahorre tener que calibrar la dirección debido al error que causa el encoder incremental al sumar pulsos. Seguidamente, se integrará el sistema de dirección en un filtro de Kalman, unificando toda la cinemática bajo un único EKF.

Más adelante, se estudiará la posibilidad de extender el método de control en aceleración y en jerk añadiendo un lazo de seguimiento de fase o PLL (del inglés phase-locked loop) para obtener transitorios más suaves. Además, se evaluará la actuación del control en velocidad usando un sistema de fusión sensorial de alto nivel que haga una actualización del Filtro de Kalman de forma asíncrona. Esta línea de trabajo va a permitir generar publicaciones para presentar en jornadas y congresos.

6.4. Valoración personal

A nivel personal, éste trabajo me ha permitido profundizar mis conocimientos en robótica móvil y perderle el miedo a la experimentación electrónica. También debo señalar la importancia del trabajo de campo en un proyecto

de estas características: para sacar el robot y obtener datos del entorno hay que dedicar unos minutos, mientras que para procesar e interpretar esos datos hay que dedicar unas horas.

Referencias

AUROVA (2018a). GitHub – AUROVA-LAB/BLUE.

AUROVA (2018b). GitHub – AUROVA-LAB/CLEAR.

Beauregard, B. (2017). Arduino Playground – PIDLibrary. [Online; consultado el 6-Julio-2018].

Berns, K. and Puttkamer, E. (2010). *Autonomous Land Vehicles: Steps towards Service Robots*. Vieweg+Teubner Verlag.

del Pino, I., Cova, S., Contreras, M. Á., Candelas, F. A., and Torres, F. (2018a). Presenting BLUE: A roBot for Localization in Unstructured Environments. In *2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 130–135.

del Pino, I., Muñoz, M. Á., Contreras, M. Á., Cova, S., Candelas, F. A., and Torres, F. (2018b). Speed estimation for control of an unmanned ground vehicle using extremely low resolution sensors. In *15th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*. Submitted for publication.

del Pino, I., Muñoz, M. Á., Cova, S., Contreras, M. Á., Candelas, F. A., and Torres, F. (2018c). Deeper in BLUE: Development of a roBot for Localization in Unstructured Environments. *Journal of Intelligent & Robotic Systems (JIRS)*, Submitted for publication.

González-Jiménez, J. and Ollero, A. (1996). Estimación de la posición de un robot móvil. *Informática y Automática. AEIA*, 29.

Koenig, N. and Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator. In *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, volume 3, pages 2149–2154. IEEE.

Ogata, K. (2010). *Ingeniería de control moderna*. Pearson Educación, Madrid.

PID controller (2018). PID controller – Wikipedia, the free encyclopedia. [Online; consultado el 6-Julio-2018].

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, volume 3, page 5. Kobe.

Robotnik Automation (2018a). Mobile robot RB-SHERPA — Robotnik. [Online; consultado el 6-Julio-2018].

Robotnik Automation (2018b). RBCAR — Robotnik. [Online; consultado el 6-Julio-2018].

Rohmer, E., Singh, S. P. N., and Freese, M. (2013). V-REP: a versatile and scalable robot simulation framework. In *Proc. of The International Conference on Intelligent Robots and Systems (IROS)*.

Shaowei, W. and Shanming, W. (2012). Velocity and acceleration computations by single-dimensional kalman filter with adaptive noise variance. *Przegld Elektrotechniczny, (2)*, pages 283–287.

Siegwart, R., Nourbakhsh, I. R., and Scaramuzza, D. (2011). *Autonomous Mobile Robots*. The MIT Press, Cambridge.

Ziegler, J. G. and Nichols, N. B. (1942). Optimum settings for automatic controller. *Transactions of ASME*, 64:759–768.